Amazon Web Services Marketplace Image

Contents

Amazon Web Services Marketplace Image User Guide

Thank you for choosing Puppet Enterprise (PE). System administrators use PE to programmatically provision, configure, and manage servers, network devices, and storage, whether in a data center or in the cloud.

This guide covers how to install, configure, and use the Puppet Enterprise Amazon Web Services (AWS) Marketplace Image.

Overview

This AWS Marketplace Image (AMI) contains a monolithic installation of PE, comprised of a collection of PE services running on a single EC2 instance—the Puppet master. The PE Architecture Overview describes PE's major services and components.

After you launch the image in EC2, PE is automatically and securely configured during operating system boot. Unique keys and certificates are generated, PE services are started, and the local node is brought under PE management.

Next, install Puppet agent on systems that will be managed by the master. Agent packages are included for all supported platforms, including *nix distributions, Windows, and Mac OS X, soyou can deploy and provision a PE managed Virtual Private Cloud (VPC).

Supported Puppet Enterprise version

Puppet offers pay-as-you-go (PAYG) and bring-your-own-license (BYOL) versions of PE as Amazon Machine Images (AMI).

For details about current PE AMI offerings, see the Amazon Web Services Marketplace. For additional information about using the PE AMI, see the PE documentation for the appropriate version.

Configuring AWS

You must run PE on an approprate EC2 instance, and manage nodes within a securely configured EC2 VPC to accomodate PE's required network ports.

EC2 instance types

You must run PE on an EC2 instance with sufficient memory and processing power.

To fulfill the hardware requirements, use an m4.xlarge instance as a minimum baseline. The PE installation contained in this image does not include additional Puppet compile masters and therefore should not manage more than 4,000 nodes.

EC2 security groups

Use this image to manage nodes within a securely configured EC2 VPC and security group. When enabling network ports for inbound connections to the Puppet master, refer to the table below, and see the EC2 security group policy example for this configuration in EC2-style JSON.

TCP port	Description	VPC Access
22	SSH	Outside VPC
443	Puppet Enterprise console (HTTPS)	Outside VPC
8140	Puppet master	Inside VPC only
8142	Orchestration services	Inside VPC only
8143	Orchestration services	Inside VPC only
61613	MCollective	Inside VPC only

For more information about PE's required network availablity, see the PE firewall configuration guide for the version of PE you are using.

Configuring the metered billing service (PAYG)

When launching the pay-as-you-go (PAYG) AMI, your EC2 instance and VPC must be configured for outbound (egress) access to the public internet, or an internet gateway allowing it to communicate with the AWS metering service. The EC2 instance must be launched with an IAM role permitted to use the metering service. For more details, see the AWS metering service guide, and the IAM role documentation.

Related information

Example EC2 security group policy on page 12
This JSON structure is an example EC2 security group policy that accommodates inbound network ports required by Puppet.

Launching the image

Launching the image ensures the Puppet master is ready to manage nodes, and for agents to connect with it.

Before you begin

Before launching the Puppet Enterprise Marketplace image, select the AMI in the AWS Marketplace, review licensing terms, and subscribe.

1. Launch an EC2 instance from the AWS console or any of a variety of AWS SDKs and third-party tools. Select or create each of the following:

   - **EC2 AMI** (the PE Marketplace image selected above)
   - **EC2 instance type** (see recommendations)
   - **EC2 VPC and subnet**
   - **EC2 security group** (see configuration details)

   To control access to the instance, the AWS console creates a new EC2 key pair. Other tools allow also allow you to use an existing key pair.
2. Connect to the EC2 instance by using the key pair created in step 1 and the username `puppetadmin` by running:

   ```
   ssh -i ~/.ssh/<EC2-KEYPAIR-PRIVATE>.pem puppetadmin@<EC2-PUBLIC-HOSTNAME>
   ```

   SSH keys are automatically provisioned by EC2, and no password is required.
3. Wait for PE configuration. which begins automatically while booting the EC2 instance. It takes about 8 minutes to complete, and must finish before you connect and manage nodes. To determine when the PE services are fully configured, run the `check_status.sh` script:

   ```
   /opt/puppetlabs/aws/bin/check_status.sh --wait
   ```
4. As root, set the console password. Console access is disabled until the password is set.

   ```
   puppet infrastructure console_password
   ```

   Tip: You can run this command at any time to reset the console password.
5. Using a web browser, connect to the console at `https://<EC2-PUBLIC-HOSTNAME>`, accept the console's certificate, and login with username `admin` and the password set in step 4.

   Since you or another administrator at your site is in full control of which certificates the Puppet certificate authority signs, the authority verifying the site is *you*. When your browser warns you that the certificate authority is invalid or unknown:

   - In Chrome, click **Advanced**, then **Proceed to <CONSOLE ADDRESS>**.
   - In Firefox, click **Advanced**, then **Add exception**.
   - In Internet Explorer or Microsoft Edge, click **Continue to this website (not recommended)**.
   - In Safari, click **Continue**.

   CAUTION: Safari certificate handling may prevent console access. Due to Apache bug 53193 and the way Safari handles certificates, you should avoid using Safari to access the PE console.

   Once you've logged in, the console indicates that the master is actively managed by showing `1 Nodes run in enforcement`; the node it refers to is the master itself.
6. Optional. Configure PE certificate auto-signing.
7. Optional. The `puppetadmin` user's password expires 60 days after the image is created. If this password is not reset, the account expires and cannot be used to log in to the image. To prevent the password from expiring, run `chage -E -1 puppetadmin` on the master.

The Puppet master is ready to manage nodes, and for agents to connect with it.

Connecting agents

To connect agents to the master, this image is configured to use private (internal) EC2 hostnames.

Before you begin

To manage nodes outside of EC2 or across EC2 VPCs, such as when managing on-premises nodes or using EC2 classic mode, configure the master by running the `update_agent_repos.sh` script:

```
sudo /opt/puppetlabs/aws/bin/update_agent_repos.sh public
```

This image contains agent packages for all supported platforms. These instructions are a quick reference to *nix agent installation and configuration process. For detailed information, including installing on other operating systems, see the supported Puppet Enterprise versions topic.

To manage a new *nix node:

1. Launch the node, then connect (SSH) to it.
2. Install the agent package for a typical *nix platform by running:

   ```
   curl -k https://<PUPPET-MASTER-HOST>:8140/packages/current/install.bash |
   sudo bash
   ```

   Use the EC2 private hostname or IP address of the master for `<PUPPET-MASTER-HOST>`.

   During installation, the agent submits a certificate signing request to the master.
3. If certificate auto-signing is not enabled on the master, manually sign the agent certificate request using the console or command line by running:

   ```
   sudo puppet cert list
   sudo puppet cert sign <NAME>
   ```

Related information

Certificates and DNS configuration on page 6
Using the master's *private* EC2 hostname, PE generates certificates which include the master's *public* EC2 hostname and `puppet` as alternate DNS names.

EC2 hostname or IP address (private vs. public address) is wrong on page 10
To communicate from managed nodes to the master, this PE installation and security group settings are configured to use the private (internal) EC2 hostname.

Supported Puppet Enterprise version on page 4
Puppet offers pay-as-you-go (PAYG) and bring-your-own-license (BYOL) versions of PE as Amazon Machine Images (AMI).

Certificates and DNS configuration

Using the master's *private* EC2 hostname, PE generates certificates which include the master's *public* EC2 hostname and `puppet` as alternate DNS names.

For more information about EC2 hostnames, see the EC2 hostname or IP address troubleshooting topic.

Managing EC2 nodes by their private hostname (rather than the public hostname) keeps their hostnames consistent if, for example, the node is resized or changed to a different EC2 instance type. This requires less work when administering a PE managed VPC.

Related information

EC2 hostname or IP address (private vs. public address) is wrong on page 10
To communicate from managed nodes to the master, this PE installation and security group settings are configured to use the private (internal) EC2 hostname.

Changing the master's hostname and regenerating certificates

While not recommended, you can change the hostname of your Puppet master and use this hostname to generate new PE certificates. Do this before you connect any agents to the master.

Note: These instructions are specific to the PE installation contained in this AMI and include some minor variations and simplifications of the instructions outlined in the PE documentation.

1. Connect to the master by running:

```
ssh -i ~/.ssh/<EC2-KEYPAIR-PRIVATE>.pem puppetadmin@<EC2-PUBLIC-HOSTNAME>
```

2. Wait for PE configuration to complete and run the `check_status.sh` script to confirm its status:

```
/opt/puppetlabs/aws/bin/check_status.sh --wait
```

3. Stop all PE services by running:

```
sudo /usr/local/bin/puppet resource service puppet ensure=stopped
sudo /usr/local/bin/puppet resource service pe-puppetserver ensure=stopped
sudo /usr/local/bin/puppet resource service pe-activemq ensure=stopped
sudo /usr/local/bin/puppet resource service mcollective ensure=stopped
sudo /usr/local/bin/puppet resource service pe-puppetdb ensure=stopped
sudo /usr/local/bin/puppet resource service pe-postgresql ensure=stopped
sudo /usr/local/bin/puppet resource service pe-console-services
 ensure=stopped
sudo /usr/local/bin/puppet resource service pe-nginx ensure=stopped
sudo /usr/local/bin/puppet resource service pe-orchestration-services
 ensure=stopped
sudo /usr/local/bin/puppet resource service pxp-agent ensure=stopped
```

4. Copy the SSL certificate directory (`/etc/puppetlabs/puppet/ssl/`) to a backup location. Should anything go wrong during this process, you can restore certificates and your PE installation.

```
sudo mv /etc/puppetlabs/puppet/ssl /etc/puppetlabs/puppet/ssl.backup
```

5. Delete the local cached catalog, which will be invalidated by the new hostname, by running:

```
sudo rm -f /opt/puppetlabs/puppet/cache/client_data/catalog/*
```

6. Set the Puppet master's new hostname. This depends on your configuration, and could be as simple as following these instructions, or this might entail configuring a DNS service like AWS's Route 53.
   a. Set the hostname: `sudo hostnamectl set-hostname <NEW-MASTER-HOSTNAME>`
   b. Add the hostname to `/etc/hosts`
   c. Add `preserve_hostname: true` to the main section of `/etc/cloud/cloud.cfg`, for example, immediately below `disable_root: 1`
7. Verify that the master and agents can resolve the new hostname. Puppet must be able to contact this hostname to connect to PE services and complete the certificate generation process.

```
ping <NEW-MASTER-HOSTNAME>
```

8. Edit the master's `/etc/puppetlabs/puppet/puppet.conf` file and set the `certname` parameter in both the `[main]` and `[master]` sections to the new hostname.

   Note: For best compatibility, limit the `certname` to letters, numbers, periods, underscores, and dashes.
9. Optional. To also include alternate DNS names, edit `/etc/puppetlabs/enterprise/conf.d/pe.conf` and set `pe_install::puppet_master_dnsaltnames` to a list of desired alternate hostnames.

   Note: If you want to change the alternate DNS names on the master later, you must repeat all of these steps.

10. Remove the contents of the config files so Puppet can regenerate them with the new hostname:

```
echo '' > /etc/puppetlabs/nginx/conf.d/proxy.conf
echo '' > /etc/puppetlabs/nginx/conf.d/http_redirect.conf
echo '' > /etc/puppetlabs/puppetdb/certificate-whitelist
echo '' > /etc/puppetlabs/console-services/rbac-certificate-whitelist
echo '<beans></beans>' > /etc/puppetlabs/activemq/activemq.xml
```

11. Remove the old hostname from `/etc/puppetlabs/puppet/autosign.conf`.
12. Use the Puppet Enterprise module to regenerate certificates and restart PE services. (The `--no-recover` and `--modulepath` options are required.)

```
sudo /usr/local/bin/puppet infrastructure configure --no-recover --
modulepath /opt/puppetlabs/server/data/enterprise/modules
```

13. Remove the former master hostname from the list of PE managed nodes by running:

```
sudo /usr/local/bin/puppet node purge <FORMER-MASTER-HOSTNAME>
```

14. Start a local agent run on the master by running:

```
sudo /usr/local/bin/puppet agent -t
```

15. To confirm the master's `certname`, run:

```
sudo /usr/local/bin/puppet config print certname
```

For more information about parameters for configuring and tuning the Puppet master, see the supported PE versions topic. Refer to the PE configuration settings for the PE version you are currently using.

Related information

Supported Puppet Enterprise version on page 4
Puppet offers pay-as-you-go (PAYG) and bring-your-own-license (BYOL) versions of PE as Amazon Machine Images (AMI).

Tuning and configuration

The PE installation contained in this image is pre-configured and tuned for use with up to 4,000 nodes.

For a list of default settings, and instructions for changing these settings and reconfiguring PE services, see tuning monolithic installations in the supported Puppet Enterprise version topic.

Related information

Supported Puppet Enterprise version on page 4
Puppet offers pay-as-you-go (PAYG) and bring-your-own-license (BYOL) versions of PE as Amazon Machine Images (AMI).

Migrating to a larger instance type

As your PE-managed infrastructure grows, moving to a larger AWS instance type improves your system performance.

The PE AMI is EBS-backed and hardware virtualized (HVM), so the resizing process doesn't require reconfiguration of PE services. Follow the AWS instructions for Resizing an EBS-backed instance.

During EC2 resizing, your instance's public hostname and IP address might change. To access the PE console, connect to the new public hostname. Resizing the instance doesn't change the internal (private) hostname or IP address, and therefore no change is required for PE services or managed nodes. See the Certificates and DNS configuration section for more information.

Related information

Certificates and DNS configuration on page 6
Using the master's *private* EC2 hostname, PE generates certificates which include the master's *public* EC2 hostname and `puppet` as alternate DNS names.

Upgrading Puppet Enterprise

There is no automated process for upgrading to newer versions of the PE AMI. As with any PE installation, however, you can manually upgrade PE to the latest release from the Puppet master's command line.

For information, see upgarding PE: monolithic in the supported PE version topic.

Related information

Supported Puppet Enterprise version on page 4
Puppet offers pay-as-you-go (PAYG) and bring-your-own-license (BYOL) versions of PE as Amazon Machine Images (AMI).

Troubleshooting

When using the Puppet Enterprise Marketplace Image, you might encounter some problems that this troubleshooting section can address.

For additional troubleshooting information, see troubleshooting in the supported PE version topic.

Related information

Supported Puppet Enterprise version on page 4
Puppet offers pay-as-you-go (PAYG) and bring-your-own-license (BYOL) versions of PE as Amazon Machine Images (AMI).

EC2 security group isn't configured correctly

This image requires a specific network configuration to allow managed agent nodes to access services on the Puppet master. For more information, see the AWS configuration instructions and the JSON-formatted example EC2 security group policy.

Related information

Example EC2 security group policy on page 12
This JSON structure is an example EC2 security group policy that accommodates inbound network ports required by Puppet.

Configuration error appears in a PAYG instance’s message of the day

If your pay-as-you-go (PAYG) EC2 instance is not properly configured to access the AWS metered billing service on launch, the instance will add an error message to the server message of the day (`/etc/motd`) and display it when you log in. Puppet Enterprise will remain in a partially-configured and non-functional state.

Correct the misconfiguration and launch a new EC2 instance.

After 60 days, the `puppetadmin` user account stops working

The default `puppetadmin` user’s password expires 60 days after the image is created. If you fail to reset the password, the account expires.

To prevent the password from expiring, run `chage -E -1 puppetadmin` on the Puppet master.

PE console password hasn't been set, or I don't have the password

Access to the PE console is disabled until you set the console password.

Run the included `set_console_password.sh` script described in the "Launching the image" section to set or reset the password.

Related information

Launching the image on page 4
Launching the image ensures the Puppet master is ready to manage nodes, and for agents to connect with it.

Agents can't connect because PE configuration isn't yet complete

When first booting the EC2 instance, PE configuration must complete before you can connect and manage agents.

Run the `check_status.sh` script described in the "Launching the image" section to confirm when configuration has finished.

Related information

Launching the image on page 4
Launching the image ensures the Puppet master is ready to manage nodes, and for agents to connect with it.

SSH username or credentials don't work when connecting to the EC2 instance

This image uses cloud-init to provision an SSH key for the *`puppetadmin`* user. AWS Marketplace policy requires root SSH access to be disabled, so users must specify an EC2 key pair when launching the EC2 instance and connect with the matching private key.

For example, you can run:

```
aws ec2 run-instance --key-name <EC2-KEYPAIR-NAME> …
…
ssh -i ~/.ssh/<EC2-KEYPAIR-PRIVATE>.pem puppetadmin@<EC2-PUBLIC-HOSTNAME>
```

Learn more about cloud-init.

EC2 hostname or IP address (private vs. public address) is wrong

To communicate from managed nodes to the master, this PE installation and security group settings are configured to use the private (internal) EC2 hostname.

To get the local hostname on the Puppet master, use Facter by running:

```
$ facter ec2_metadata.local-hostname

ip-….compute.internal
```

Or retrieve the EC2 instance's metadata by running:

```
$ curl http://169.254.169.254/latest/meta-data/local-hostname

ip-….compute.internal
```

To access the PE console from outside EC2, use the Puppet master's public EC2 hostname:

```
$ facter ec2_metadata.public-hostname

ec2-….compute.amazonaws.com

$ curl http://169.254.169.254/latest/meta-data/public-hostname

ec2-….compute.amazonaws.com
```

Then use your web browser to connect to `https://ec2-…compute.amazonaws.com`. See the certificates and DNS configuration section for more information about EC2 hostnames and the default DNS configuration.

Related information

Certificates and DNS configuration on page 6

Using the master's *private* EC2 hostname, PE generates certificates which include the master's *public* EC2 hostname and `puppet` as alternate DNS names.

Puppet agent run won't work when initiated by a non-root user

An agent run initiated by `puppetadmin` or any other non-root user will fail when attempting to access certificates, packages, and services.

Always start manual Puppet agent runs with super-user privileges:

```
sudo /usr/local/bin/puppet agent -t
```

Other issues

Refer to PE documentation when troubleshooting any of the following Puppet or PE features.

- Connections
- Code Manager
- Databases
- Puppet core
- MCollective
- Windows

Related information

Supported Puppet Enterprise version on page 4
Puppet offers pay-as-you-go (PAYG) and bring-your-own-license (BYOL) versions of PE as Amazon Machine Images (AMI).

Support

You can get help with Puppet from us and the rest of the Puppet community. If you've purchased Puppet Enterprise, you can also access our knowledge base and open a support ticket.

Bringing (or buying) your own license

You can bring your own license (BYOL) for Puppet Enterprise 2017.2 or 2017.1, and use it in the Amazon Marketplace image. This can be a license you already own or one purchased to use with AWS — PE licenses are transferrable between nodes, whether they're bare-metal, virtual machines, or in the cloud. Unlike the pay-as-you-go (PAYG) option, this licensing method doesn't require the server to have access the public internet for billing.

When your PE purchase is complete, you'll receive a login to our support portal, where you can search the knowledge base or open a case for support.

Free Trial

If you're evaluating Puppet Enterprise, contact Sales for support.

You can also get help from Puppet community members (and Puppet employees who drop in to lend a hand) in these channels:

- ask.puppet.com
- The puppet-users Google Group
- The #puppet and #puppet-enterprise channels on Puppet Community Slack
- The #puppet channel via IRC on Freenode

Also, consult our community guidelines for more resources and advice on asking for and providing help in our community.

Paying as you go

With the Pay As You Go (PAYG) licensing option, you pay for Puppet Enterprise 2017.2 hourly and by the number of managed nodes using the AWS metered billing service instead of a standalone PE license. The PAYG AMI monitors the number of nodes being managed by PE and submits this data hourly to the AWS metered billing service, which charges you for what you use. This is more flexible and easier to start than the BYOL option — there's no license purchasing process or management, and billing is handled in your existing AWS account.

Pricing depends on the number of managed nodes. For the first 10 nodes (including the Puppet master), you pay only for the underlying EC2 infrastructure — the PE license for these nodes is free. For pricing beyond 10 nodes, see the Amazon Marketplace page. Your AWS billing statement show your total usage and cost.

To monitor usage, the PAYG AMI includes a service installed along with Puppet Enterprise that periodically queries the PuppetDB service for the number of unique Puppet agents that have checked in during the last hour. (By default, Puppet agents check in to the master every 30 minutes.) The service log, located at `/var/log/puppetlabs/pe-ami.log` on the master, records each usage sample and all billing data sent to the AWS metering service.

Because of this AWS metering service requirement, a PAYG master must be able to access the public internet. For details on configuring your instance and VPC, see the AWS configuration instructions.

Example EC2 security group policy

This JSON structure is an example EC2 security group policy that accommodates inbound network ports required by Puppet.

```
{
    "IpPermissions": [
      {
          "PrefixListIds": [],
          "FromPort": 22,
          "IpRanges": [{"CidrIp": "0.0.0.0/0"}],
          "ToPort": 22,
          "IpProtocol": "tcp",
          "UserIdGroupPairs": []
      },
      {
          "PrefixListIds": [],
          "FromPort": 443,
          "IpRanges": [{"CidrIp": "0.0.0.0/0"}],
          "ToPort": 443,
          "IpProtocol": "tcp",
          "UserIdGroupPairs": []
      },
      {
          "PrefixListIds": [],
          "FromPort": 8140,
          "IpRanges": [{"CidrIp": "<SUBNET-CIDR>"}],
          "ToPort": 8140,
          "IpProtocol": "tcp",
          "UserIdGroupPairs": []
      },
      {
          "PrefixListIds": [],
          "FromPort": 8142,
          "IpRanges": [{"CidrIp": "<SUBNET-CIDR>"}],
          "ToPort": 8142,
          "IpProtocol": "tcp",
          "UserIdGroupPairs": []
      },
      {
          "PrefixListIds": [],
```

```
            "FromPort": 8143,
            "IpRanges": [{"CidrIp": "<SUBNET-CIDR>"}],
            "ToPort": 8143,
            "IpProtocol": "tcp",
            "UserIdGroupPairs": []
        },
        {
            "PrefixListIds": [],
            "FromPort": 61613,
            "IpRanges": [{"CidrIp": "<SUBNET-CIDR>"}],
            "ToPort": 61613,
            "IpProtocol": "tcp",
            "UserIdGroupPairs": []
        }
    ],
    "IpPermissionsEgress": [
        {
            "IpProtocol": "-1",
            "IpRanges": [{"CidrIp": "0.0.0.0/0"}],
            "UserIdGroupPairs": [],
            "PrefixListIds": []
        }
    ]
}
```